C’est quoi data Mining ?

Data Mining c’est une cherche d'information avancée

C’est quoi Big Data ?

Big data c'est une grand quantite de donne complexes nécessitant des outils spéciaux pour être stockées,traitées et analysées

Relation entre data Mining et Big Data?

Big Data Fournit les donneés et les entils, tandis que data mining utilise des techniques pour en extraire des Connaissances

Relation entre Data Mining et Analyse des donnés

L’analyse des donneés utilise des méthodes statistique pour Comprendre les donnés, tandis que le data mining extrait des modèles cashés à l'aide d’algorithmes avancés

Relation entre Data mining et Intelligences artificielle?

Le DM utilise des techniques d'IA, comme l'apprentissage automatique pour extraire des modéles cachés et améliorer L'analyse des donneés

Méthodes et techniques de DM pour L'exploration des données?

classification, clastering, association regression

**2. Importance de Python dans la scienc** 1 .Langage Polyvalent : Python est facile à apprendre et offre

une syntaxe intuitive, idéale pour les débutants comme pour

les experts.

2. Écosystème Riche : Bibliothèques populaires comme

Pandas, NumPy, Matplotlib, Scikit-learn et TensorFlow pour

l’analyse, la visualisation et l’apprentissage automatique.

3. Communauté Active : Large support communautaire avec

des forums, des tutoriels et des ressources pour résoudre les

problèmes.

4. Intégration Facile : S'intègre avec d'autres technologies et

langages pour un traitement des données à grande échelle.

5. Utilisation Industrielle : Standard dans les entreprises pour

l’analyse des données, le Big Data et l’intelligence artificielle.

6. Puissance et Flexibilité : Gestion efficace des tâches

complexes comme la manipulation de données massives, la

modélisation et la visualisation interactive.

**Vue d'ensemble du pipeline d'exploration de données :**

1.Sélection : Choix des données pertinentes dans un ensemble de données plus large.

2.Prétraitement : Préparation des données sélectionnées pour l'analyse.

3.Transformation : Conversion des données prétraitées dans un format adapté au Data Mining.

4.Data Mining : Analyse des données transformées pour découvrir des motifs.

5.Interprétation/Évaluation : Interprétation et évaluation des motifs découverts pour extraire des

connaissances significatives.

**Outils Python pour la collecte de données**

1.Pandas : Manipulation et analyse des données. Lecture des fichiers CSV, Excel, SQL.

2.BeautifulSoup : Analyse HTML et XML pour le scraping web. 3.Requests : Récupération de données via APIs et services web.

4.Connecteurs SQL : mysql-connector-python, psycopg2, sqlite3.

5.Yahoo Finance : Extraction de données financières (prix, historiques).

Collecte des données avec Python

1.Pourquoi utiliser différents formats de fichiers ?

- Assurer la compatibilité avec divers outils et applications.

2.Formats populaires et outils Python associés :

-CSV : Lecture simple et largement supportée (pandas.read\_csv()).

-TXT : Pour textes simples (pandas.read\_csv() avec délimiteurs personnalisés).

-XLSX : Pour données complexes avec feuilles multiples (pandas.read\_excel()).

-JSON : Données structurées, faciles à lire (pandas.read\_json()).

-XML : Pour données hiérarchiques (pandas.read\_xml()).

-HTML : Extraction de tableaux web (pandas.read\_html())

**Exemple d'utilisation avec CSV**

Utiliser Pandas pour lire un fichier CSV

import pandas as pd

df = pd.read\_csv('data.csv’)

print(df.head())

Avantages de Pandas :

 Détection automatique des délimiteurs.

 Gestion des valeurs manquantes.

 Analyse et manipulation rapides via DataFrames.

**Données JSON et XML**

Manipulation de JSON et XML

JSON :

 Format d’échange léger.

 Lecture avec Pandas :

XML :

 Données balisées.

 Extraction avec Pandas :

Applications : API REST, stockage de configurations.

df = pd.read\_xml('data.xml’)

df = pd.read\_json('data.json’)

**Collecte depuis une base SQL**

Connexion aux bases de données SQL

Outils populaires :

\* mysql-connector-python.

\* psycopg2 (PostgreSQL).

\* sqlite3 (bases locales).

import sqlite3

conn = sqlite3.connect('database.db')

query = "SELECT \* FROM table\_name"

df = pd.read\_sql\_query(query, conn)

**Conclusion et cas pratiques**

 La collecte de données est une étape clé du pipeline d’analyse.

 Python offre une panoplie d’outils pour gérer des données de sources

variées.

 Prochaines étapes :

 Études de cas sur le nettoyage et l’analyse.

 Mise en œuvre d’un projet comple

